Visual divisive hierarchical clustering using k-means

Matic Perovšek[1], Nada Lavrač[1,2], Bojan Cestnik[1,3]

[1] Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia
[2] University of Nova Gorica, Nova Gorica, Slovenia
[3] Temida d.o.o., Ljubljana, Slovenia
{matic.perovsek, nada.lavrac, bojan.cestnik}@ijs.si

ABSTRACT

This paper presents a browser-based semi-automatic taxonomy construction tool Vd-chuck which is able to incorporate text and data mining algorithms into a user-friendly interface. The presented system is browser-based. Its unsupervised learning for concept suggestion and different visualization techniques assist the user with textual and numerical data analysis. We tested the Vd-chuck system on a real-world domain: a corpus of documents taken from Slovenian Language technologies conferences. The results show that with our system similar taxonomies as with other taxonomy editors can be constructed.

1 INTRODUCTION

Taxonomies and ontologies have been often considered as the most adequate knowledge representation formalisms for representing the relations between different domain objects. New directions for future ontology editors have been stimulated by the rapid growth in the need of textual and numerical data analysis, growth of the internet and also by the need for mobility and collaboration. All current major ontology editors (e.g. Protege [2], OntoStudio [13], Ontogen [1]) offer little support for user collaboration and mobility, while existing browser-based editors (e.g. Web-Protoge [12], Knoodl[1]) provide only basic ontology editing functionality with barely no tools for visualization.

Vd-chuck is a browser-based taxonomy editor, offering a similar functionality as the desktop topic ontology editor Ontogen [1]. Ontogen allows users construction of new topic ontologies, as well as visualization and exploration of the existing. Vd-chuck's main advantage over Ontogen is its accessibility. Since it is browser-based, it can be accessed from anywhere at any time. The Vd-chuck system combines several data mining and text mining techniques with an intuitive user interface. Different tasks require building taxonomies differently, so it is essential to include the user's knowledge as well. Vd-chuck is semi-automatic, which means that it is able to provide suggestions–such as concept naming, concept relations, assigning examples to concepts–during taxonomy construction. Although Vd-chuck supports simplified taxonomy and concept generation, it in the end still relies on the user's background knowledge to make appropriate corrections.

The paper is organized as follows. In Section 2 we present the main components of the Vd-chuck system. As Vd-chuck was inspired by Ontogen, we provide a comparison between the two in Section 3. In Section 4 we present a real-world use case of our system on a textual domain. Section 5 concludes the paper and gives some ideas for further development.

[1] http://knoodl.com

2 OVERVIEW OF THE VD-CHUCK SYSTEM

This section describes three major components of the Vd-chuck system: the concept hierarchy, concept management and finally concept visualization.

2.1 Concept hierarchy

One of the main components of the Vd-chuck system is the concept hierarchy tab, which is always present in the upper-left side of the screen (Figure 1). It shows the structure of the taxonomy in a tree-structured way, while giving the user an option of concept selection.

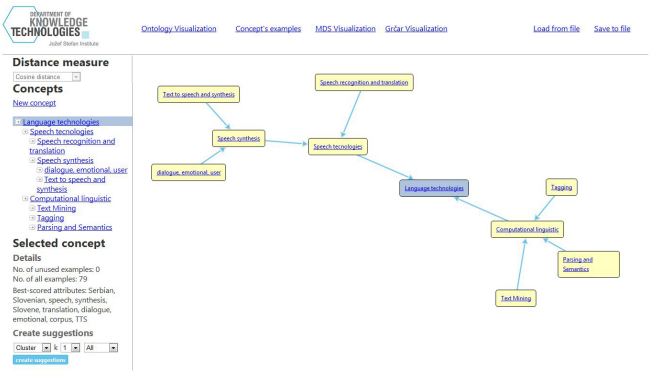

Figure 1: Vd-chuck's taxonomy visualization tab: taxonomy's concept hierarchy is displayed in the upper-left window; selected concept's details are listed on the bottom-left while the central part shows the taxonomy as a tree-structured directed graph, representing the concept hierarchy.

Under the concept hierarchy window various details of the currently selected concept are present: such as concept's name, example count, unused example count and the most informative attributes. Example count represents the number of all examples that are either part of the selected concept or in one of its descendants, while the number of unused example count represents the count of examples that are part of the selected concept but are not present in any of its descendants. The most informative attributes are the ten best scored attributes using attribute ranking method InfoGain [7], although when textual data is used ten words with the highest TF-IDF [8] values are selected.

As we can see from Figure 1, the main part of the screen presents an alternative visualization of the taxonomy—a directed graph with vertices representing the concepts and and edges representing relationship 'subconcept of'. Each concept can be selected, renamed, deleted or even moved as a subconcept to another concept by right clicking it and selecting the appropriate choice from the context menu.

The Vd-chuck system provides unsupervised learning for concept suggestion. The unsupervised learning is performed using the k-means clustering method [11], which uses one of the distance metrics available for selection with numerical data and cosine similarity [11] when using text data. Clustering can be done using the selected concept's all examples or only its unused examples. After clustering suggestions appear on the screen and the user can manually add appropriate ones to the taxonomy. Suggested concepts' initial names are generated as top three most informative features/words for that concept.

Another feature is creating concepts according to examples' classes. This feature is useful when an existing taxonomy or a dataset with a class variable needs to be altered or checked for outliers [4].

2.2 Concept management

The Vd-chuck's concept management tab is used to present all examples in correlation with the selected concept's centroid. The centroid of every concept is calculated with the k-means method as the average value for numerical attributes and most frequent value for discrete. The distance used to calculate the centroid depends on type of the data; when using textual datasets, cosine similarity is used. When working with numerical data, one of the following distances can be selected as the dissimilarity measure: Euclidean, Manhattan, Relieff, Hamming or PearsonR distance.

As shown in Figure 2.2, every example along with its unique identifier and distance from the selected concept's centroid is listed in the main window. Examples that belong to the selected concept (or one of its descendants) are marked with a different color. When clicked upon, each feature (or textual content for textual datasets) of the example is listed on the right side of the main window.

A similarity graph of all examples with selected concept's centroid is shown on the bottom of the main window. Each

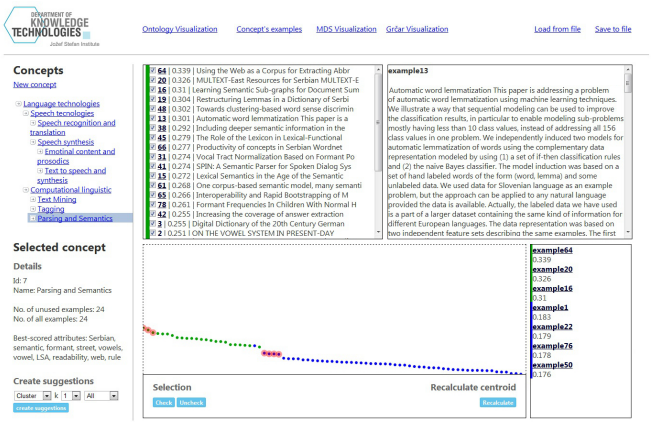

Figure 2: The concept management tab. Examples are listed in the top middle, while the selected example's content is shown on the right. Similarity graph of examples with concept's centroid is drawn on the bottom of the screen.

example is presented with a dot, colored according to its belonging to the selected concept. The similarity graph is very useful for outlier detection. The user can look at each example and decide upon moving it to another (more suitable) concept based on his own background knowledge. This can be done by simply dragging it to the desired concept's name on the always-present concept hierarchy on the left.

2.3 Concept visualization

The problem of projecting multidimensional data into two dimensional space has been investigated by different researchers due to its potential application to data analysis. As visualization is a useful tool for gaining insights into overwhelming amounts of data, the Vd-chuck system provides users with two different types of visualisation: multidimensional scaling MDS [6] and a MDS-like distance-preserving projection onto a 2D canvas, presented in [3].

MDS. MDS visualization uses dissimilarities between pairs of different examples. As mentioned, in the Vd-chuck system the user can choose between several similarity measures. Vd-chuck uses Sammon's projection [9] for mapping high-dimensional spaces to spaces of lower dimensionalities. Sammon's projection tries to preserve the structure of inter-point distances in high-dimensional spaces in the lower-dimension projections. This is done by minimizing the error function, which is often referred to as Sammon's stress:

$$E = \frac{1}{\sum_{i<j} d^*_{ij}} \sum_{i<j} \frac{(d^*_{ij} - d_{ij})^2}{d^*_{ij}}, \qquad (1)$$

where variable d^*_{ij} denotes the distance between i-th and j-th objects in the original space, while d_{ij} is the distance between their projections. The minimization of Equation 1 is a rather complex problem that cannot be solved in closed-form. Therefore, MDS algorithms use iterative numerical al-

gorithms to find a matrix that minimizes the stress function. Our implementation uses the SMACOF algorithm [5], which is based on iterative majorization.

Although Sammon's projection can unfold data belonging to manifolds of high-dimension, once large distances are taken into account in the optimization, it can fail for highly twisted spaces. As a result we see very similar cases projected far apart. A serious problem of Sammon's projection is also its relatively high computational complexity $O(n^2)$.

Document stream visualization. The method presented in [3] is a pipeline of different data mining techniques aimed at better mapping of higher-dimensional data onto a 2D canvas. First, the algorithm clusters examples in order to segment the data space. Using the k-means clustering method several (usually 100) centroids are obtained. These centroids are used as control points in the final stage of the visualization pipeline. Second, these representative instances are projected onto a planar canvas by using an iterative stress majorization method. Simultaneously, determining nearest neighbours for every point is required. Cosine similarity is used for computation of distances between examples. Finally, the least-squares solver is used to interpolate non-control points between the coordinates of control points. The basic idea how to construct the system of linear equations required by the interpolation process is that each (control or non-control) point can be described as the center of its nearest neighbors. The result of the solver is a n-dimensional vector which contains pairs of coordinates for every example.

Visualization tools. Vd-chuck's visualizations provide the user with different tools for easier interpretation of the visualized concept. Every instance can be selected by simply clicking on it (note that multiple area selection is possible too). Selected examples are listed on the right of the visualization window.

The 'Highlight selected on similarity graph' is used to check the similarity of all of the selected examples on the visualized concept's similarity graph. This is mostly useful for outlier detection and rearranging examples to other possibly more suited concepts. The 'Compare selected examples with concept's centroid' function gives the user a possibility to check how selected examples compare to the concept's average values, while the 'Calculate most influential attributes for selected examples' provides the user a list of attributes that best differentiate the selected examples against other examples of the visualized concept. The most influential attributes are obtained with the Relief measure.

3 COMPARISON WITH ONTOGEN

Vd-chuck was inspired by the desktop topic ontology editor Ontogen. In this section we provide a direct comparison between the two. The comparison is carried out according to four characteristics: browser-base, handling of numerical and textual data, active learning and visualization techniques.

Browser-based. As opposed to Ontogen, the Vd-chuck system is browser-based, so it can be accessed from anywhere. It does not need preinstalled software except for the web-browser. It is always up-to-date, so there is no need for local version upgrades. It is also possible to access it from mobile phones and tablets. Unlike in Ontogen, the user is not required to save and transfer files locally. Consequently, it enables easier collaboration as users can share their projects by simply sharing a link of the Vd-chuck's project web page.

Handling numerical or textual data. Vd-chuck offers taxonomy construction on either numerical or textual data, while Ontogen can deal only with textual data. When loading textual data in Vd-chuck, some basic text preprocessing (such as lemmatization and stop-word removal) and a TF-IDF transformation of all documents is done. TF-IDF features are then used in clustering and visualization algorithms.

Active learning. Vd-chuck system does not provide the supervised SVM active learning method [1] present in Ontogen. The user is therefore required to manually move each example that in his opinion does not suit the selected concept. When much example moving is required the absence of active learning can slow down the taxonomy construction process.

Visualization. Vd-chuck offers two visualization techniques: MDS and the document stream visualization. In contrast with Ontogen's, Vd-chuck's visualizations provide background colouring of examples according to their belonging concept. Meanwhile, Ontogen provides an option of mapping document keyword onto the 2D space. Furthermore, the density of documents in an area is used for generating the background relief in Ontogen.

4 TYPICAL VD-CHUCK USE-CASE

In this section we describe a real-life use case of Vd-chuck. We show how to build a topic ontology from textual data, in order to do so we repeated the experiment described in [10]. Construction of a topic ontology of a corpus of 79 English-written documents taken from Language technologies conferences, held in Ljubljana from 1998 to 2010, was performed. All documents were previously preprocessed—data transformations such as discarding authors' names, institutions, references, footnotes and page numbers were performed.

We used Vd-chuck's concept suggestion tool for every concept with different k values for the k-means clustering algorithm. The used k value was the one which splits the data to most sensible big-enough clusters, confirming the user's understanding of the area with its keywords. If a concept could not be split into reasonable subconcepts further concept division was not performed.

We decided to use a top-down approach of taxonomy generation. We started by dividing the root concept into two subconcepts. The Vd-chuck's suggested concepts' extracted keywords were very consistent with the general division of the field of language technologies which consists of computational linguistics and speech technology. Some additional hu-

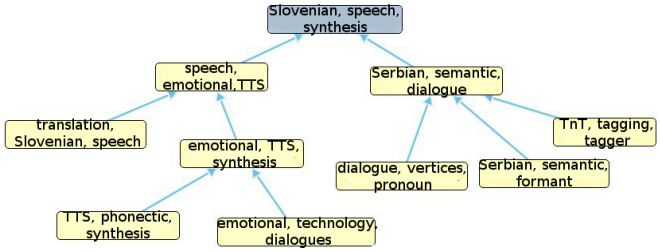

Figure 3: Taxonomy of documents from Language technologies documents before manual concept renaming. Concept names consist of most representative keywords.

man effort was also needed—some examples had to be moved to more suitable concepts.

Next, concepts of computational linguistics and speech technology where further divided. Again k-means suggestion tool was used. Inside the computational linguistics concept a general cluster (described by keywords such as 'serbian') was continually present. All examples from this cluster were manually moved according to user's knowledge and presentation of what other concepts are representing. After the sorting we could easily identify some other outliers using Vd-chuck's concept similarity graph.

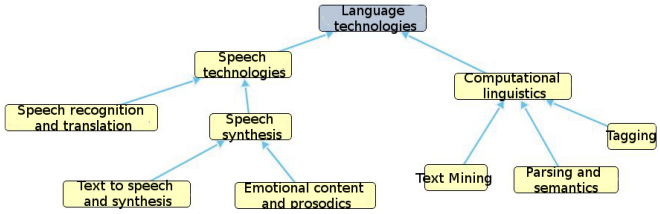

Figure 4: Updated taxonomy after manual moving of examples and concepts renaming.

Lastly, we renamed concepts in order to get a more comprehensible ontology. The result of our taxonomy generation can be seen in Figure 4. For the validation of a logical division, we checked the concepts' centroid similarity graphs. The results consistently showed that the examples in the concept were always very similar to the concept's centroid, while other examples were much further apart.

The presented topic ontology construction resulted in a similar topic ontology as described in [10]. The differences that we encounter, such as additional division of concept 'Speech recognition and translation', were mostly due to subjective judgment of the user.

5 CONCLUSION

This paper presented Vd-chuck, a browser-based semi-automatic taxonomy construction tool. We have provided a detailed comparison with the desktop ontology editor Ontogen. We have tested the Vd-chuck system on a real-life domain. The results show that with our system similar taxonomies as with other taxonomy editors can be constructed. The system is easy to use, although the lack of active learning makes taxonomy construction more time-consuming for the user. For further work we plan to construct more representative visualization, especially when dealing with smaller databases. Furthermore, work on adding active learning is planned.

References

[1] B. Fortuna, M. Grobelnik, and D. Mladenić. Ontogen: Semi-automatic ontology editor. *Procedings of Human Interface and the Management of Information. Interacting in Information Environments Conference*, pages 309–318, 2007.

[2] J.H. Gennari, M.A. Musen, and R.W. Fergerson. The evolution of protégé: an environment for knowledge-based systems development. *International Journal of Human-Computer Studies*, 58(1):89–123, 2003.

[3] M. Grčar, V. Podpečan, M. Juršič, and N. Lavrač. Efficient visualization of document streams. In *Proceedings of Discovery Science Conference*, pages 174–188. Springer, 2010.

[4] Z. He, X. Xu, and S. Deng. Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24(9):1641–1650, 2003.

[5] W.J. Heiser and I. Stoop. Explicit smacof algorithms for individual differences scaling. Technical report, PROXSCAL Progress Report, 1986.

[6] J.B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[7] J.R. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[8] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[9] J.W. Sammon Jr. A nonlinear mapping for data structure analysis. *IEEE Transactions on computers*, 100(5):401–409, 1969.

[10] J. Smailović and S. Pollak. Semi-automated construction of a topic ontology from research papers in the domain of language technologies. In *Proceedings of 5th Language & Technology Conference, Poznan*, pages 121–125, 2011.

[11] M. Steinbach, G. Karypis, V. Kumar, et al. A comparison of document clustering techniques. In *Proceedings of KDD Workshop on Text Mining Conference*, 2000.

[12] T. Tudorache, J. Vendetti, and N.F. Noy. Web-protege: A lightweight owl ontology editor for the web. *5th OWL Experiences and Directions Workshop*, 2008.

[13] M. Weiten. Ontostudioas a ontology engineering environment. *Semantic Knowledge Management*, pages 51–60, 2009.